JAPANESE PATENT APPLICATION

(11)Publication number: 09101786

(43)Date of publication of application: 15.04.1997

(21)Application number: 07258019

(71)Applicant: MATSUSHITA ELECTRIC IND CO LTD

(22)Date of filing: 04.10.1995

(72)Inventor: NOMURA NORIO

(54) MELODY GENERATING DEVICE BY DSP

[Scope of Claims for Patent]

[Claim 1] A melody generating apparatus by DSP comprising: a score data table for memorizing a score as note data; control means for performing control to play back the note data in order of being read from said score data table; tone generating means for generating a repetitive or frequency signal for a pitch period of the note as a tone signal of the note indicated by the read-out note data; envelope generating means for enveloping the tone signal; and a multiplier which multiplies outputs of said tone generating means and said envelope generating means to output a melody signal.

[Claim 2] The apparatus according to claim 1 further comprising a multiplier for multiplying the melody signal output from said multiplier by a volume signal.

[Claim 3] The apparatus according to claim 1 further comprising one or more waveform data tables used when said

tone generating means generates a tone signal.

[Claim 4] The apparatus according to claim 1 further comprising one or more envelope parameter tables used when said envelope generating means generates an envelope waveform.

[Claim 5] The apparatus according to claim 1 further comprising a note data table indicative of a pitch period for each note number.

[Claim 6] The apparatus according to claim 1 further comprising a note counter indicative of a note currently under performance in said score data table.

[Claim 7] The apparatus according to claim 1 further comprising a measure counter incremented for each sampling period so that the length of a note will be controlled.

[Claim 8] The apparatus according to any one of claims 1 through 7, wherein said score data table includes serial numbers serially assigned to notes, note numbers indicative of note names, lengths of the notes, volumes of the notes, numbers of waveform data tables used when said tone generating means generates a tone signal, and numbers of envelope parameters used when said envelope generating means generates an envelope waveform.

[Claim 9] The apparatus according to any one of claims 1 through 8, wherein said tone generating means reads out said waveform data table from top to bottom in order at each fixed interval determined by the scale of a note, so that the waveform is shifted and added at each pitch period of the note, thereby generating a repetitive or frequency waveform

having the pitch period of the scale.

[Claim 10] The apparatus according to any one of claims 1 through 9, wherein said envelope generating means generates an envelope waveform using parameters for determining rise and fall times of the envelope waveform.

[Detailed Description of the Invention]

[0001]

[Field of the Invention]

This invention relates to a melody generating apparatus using DSP for generating melodies such as a holding tone in an answering device, a cellular phone, etc.

[0002]

[Description of the Prior Art]

In recent years, DSP is equipped in an answering machine etc. The function conventionally performed by the analog signal processing has come to be performed by the digital signal processing. Then, various equipment is developed using DSP to generate melodies such as a holding tone.

[0003]

A conventional melody generating apparatus by DSP will be described below. Fig. 30 shows the general configuration of the conventional melody generating apparatus by DSP. In Fig. 30, 101 is an oscillator, which provides a clock signal to a DSP 102 for the DSP operation. 102 is the DSP as a high-speed processor that performs arithmetic processing on

data in accordance with instructions memorized in a ROM 103. 103 is the ROM, which stores coded data indicative of instructions for the DSP 102, a melody coded at a coder, and the like. 104 is a RAM for memorizing intermediate results of the arithmetic processing. 105 is an oscillator for generating a sampling frequency fs. 106 is a DA converter, which converts the digital numeric value of an input to an analog signal for every period of the sampling frequency fs. 107 is an LPF, which attenuates the signal of an fs/2 frequency component or more.

[0004]

Fig. 31 is a configuration for generation of a melody realized by the arithmetic processing in the DSP 102 of the conventional melody generating apparatus. In Fig. 31(a), 108 is a coder for coding an input melody signal, and 109 is coded data created by the coder 108 and written to the ROM 103. For example, if a 32-kbps ADPCM method is used for coding, a melody for a second will be 32-kbit coded data. In Fig. 31(b), 110 is coded data memorized in the ROM 103, and 111 is a decoder for decoding the coded data read out from the ROM 103 in accordance with an instruction from the DSP.

[0005]

The melody generating operation of the melody generating apparatus configured such above will be described below. Prior to the generation of a melody, the melody to be generated is coded at the coder 108, and the coded data 109 is created and written to the ROM. Upon performance of the

melody, the decoder 111 reads the coded data 110 memorized in the ROM 103 in order, and decodes them to output PCM digital signals to be forwarded to the DA converter 106, thus generating the melody.

[0006]

[Problem to Be Solved by the Invention]

In the conventional melody generating apparatus by DSP mentioned above, however, the coded data created by coding a melody through a voice coder or the like is memorized in a memory beforehand, and decoded when the melody is to be played back. Therefore, it has problems that large memory capacity for memorizing the coded data and large amounts of operations for decoding are required.

[0007]

The present invention has been made to solve the above-mentioned conventional problems, and it is an object thereof to provide an excellent melody generating apparatus capable of generating various timbres of a melody using DSP with small memory capacity and small amounts of operations.

[0008]

[Means to Solve the Problems]

To achieve the above-mentioned object, the present invention is to include a score data table for memorizing a score as note data, control means for performing control to play back the note data in order of being read from the score data table, tone generating means for generating a repetitive or frequency signal for a pitch period of the note as a tone

signal of the note indicated by the read-out note data, envelope generating means for enveloping the tone signal, and a multiplier which multiplies outputs of the tone generating means and the envelope generating means to output a melody signal.

[0009]

[Operation]

As described above and according to the present invention, the configuration is such that a signal of each note is generated according to the score data, so that the score data on a melody has only to be memorized in memory, thus generating the melody with small memory capacity. In addition, it does not employ the conventional coding/decoding method, so that the melody can be generated with small amounts of operations.

[0010]

[Examples]

Examples of the present invention will be described below with reference to the accompanying drawings. Although the general hardware configuration of a melody generating apparatus according to the present invention is the same as the conventional hardware configuration shown in Fig. 22, this invention differs from the conventional in memorizing to the ROM 103 instructions for generation of a melody and data tables such as the score data table, and algorisms used for operations on the DSP 102.

[0011]

(Example 1)

Figure 1 shows the configuration of the melody generating apparatus using DSP in the 1st example of this invention. In figure 1, 1 is a melody generating controller which performs the control of sequentially generating notes memorized in a score data table 5. 2 is a measure counter which increments a value mc for every sampling period to measure a note length. 3 is a note counter which counts a value nc indicating a note of the score data table 5 currently under performance. 4 is a score data table read-out device, and reads the note data nno, leng, gain, sno, and eno of a note indicated by the value nc of the note counter 3 from the score data table 5. The score data table 5 will be described later. 6 is a note data table read-out device, and looks for a note having a note number nno from a note data table 7. The note data table read-out device 6 reads Tp and Tw, which are parameters of the note from the note data table 7. 8 and 9 are waveform tables and store the various waveforms used for tone generating. 10 is a waveform selector and selects a waveform by waveform No. sno from the waveform tables 8 and 9. The selected waveform is used by a tone generator 11 for tone generating. 11 is the tone generator (SG), uses the data of the waveform tables 8 and 9, and generates a frequency signal x of a pitch period indicated by Tp and Tw. 12 is an envelope parameter table and stores various kinds of envelope parameters. 13 is an envelope pattern selector, which selects, from the envelope

parameter table 12, envelope parameters AR, DiR, D2R, RR, DiL, and LIM, which are used in case an envelope generator 14 generates an envelope waveform. 14 is the envelope generator (EG), which generates an envelope waveform e by the selected envelope parameters AR, DiR, D2R, RR, DiL, and LiM. 15 and 16 are multipliers, which perform the multiplication of x, e, and gain, and output a melody signal y.

[0012]

Fig. 2 shows a form of the score data table 5. The score data table 5 stores the score of a melody to be generated in which a line of data represents a note. The notes are played back in order from the top. The data table includes the following items:

Serial number: Numbers serially assigned to notes

Note name: Names of the notes

Note No. nno: Note numbers representing the notes assigned respective note names

Length leng: Lengths of the respective notes

Gain gain: Volumes of the respective notes

Waveform data No. sno: Numbers of waveform data tables used when the tone generator 11 generates a tone signal

Envelope No eno: Numbers of envelope parameters used when the envelope generator 13 generates an envelope waveform

The timbre of a note is determined by the waveform No. sno and the envelope No. eno.

[0013]

Fig. 3 shows a concrete example of the score data table

5 for a score shown in Fig. 4. For example, serial number 1, note name G5, note No. 32 and length 24 designate a half note in the fifth octave G; serial number 31, note name **, note No. 38 and length 12 designate a quarter rest, and serial number 63, note name ED, note No. 39 and length 0 designate a full stop indicating the end of a piece of music. In other words, the last line of the score data table must be note name ED and note No. 39. In addition, since the length of a measure is 48 in this example, the value of the length leng is 48 for a whole note, 24 for a half note, and 12 for a quarter note. Although the note names are stored in the score data table 5 in this example for purposes of illustration, only the note Nos. are required and the note names are not necessary in actual situations.

[0014]

Fig. 5 shows a form of the note data table 7. A line of data represents a note with a note name, Tp, and Tw stored for each note No. Values of Tp and Tw represent pitch information on the note; they are used as parameters when the tone generator 11 generates a tone signal.

[0015]

Fig. 6 shows a concrete example of the note data table 7. For example, note No. 32 indicates that the note name is G5, and the pieces of pitch information Tp and Tw are 653 and 1606, respectively. Note No. 38 and note name ** represent a quarter note rest, and note No. 39 and note name ED represent the end of the piece. Although the note names are stored in

the score data table 7 in this example for purposes of illustration, they are not necessary in actual situations.

[0016]

Fig. 7 shows the waveform data tables 8, 9. The waveform data is used when the tone generator 11 generates a repetitive or frequency waveform as a tone signal x. In Fig. 7, there are Sn waveform data tables.

[0017]

Fig. 8 shows an example of the envelope parameter table 12. The table stores the value of an envelope with envelope parameters AR, D1R, D2R, RR, D1L, and L1M on a line basis. These parameters are used as parameters when the envelope generator 13 generates an envelope waveform e. In Fig. 8, there are three sets of envelope parameters.

[0018]

Fig. 9 shows an algorism used in the melody generating controller 1 for generating a melody in the example. The operation of the algorism will be described below in correspondence with the block diagram of Fig. 1.

ST1) Initialization is done.

Initialization is done as follows:

Measure counter mc = 0, note counter NC= 1, and ending flag = 0, where the ending flag is a flag that becomes 1 upon completion of performance of the score data table 5 from top to bottom.

ST2) The ending flag is checked.

This step branches to ST8 when ending flag ≠ 0.

ST3) An operation command CMD is set for the tone generator (SG) 13 and the envelope generator (EG) 14.

The operation command CMD and parameters to be given to the SG and EG are set according to the value mc. First, the value mc is checked:

mc == 0 causes a branch to ST3.1;

mc == leng * Temp - 1 causes a branch to ST3.2; and

mc < mc < leng * Temp - 1 causes a branch to ST3.3,

where Temp is a constant indicative of the tempo of the piece of music. If the length of a quarter note is T, Temp = (fs x 4)/T x L, where

fs is a sampling frequency [Hz], and

L is leng indicative of the length of a measure (L = 48 in the score data table of Fig. 2).

[0019]

ST3.1) When mc == 0, a new note signal is generated.

ST3.1.1) Score data is read out.

The score data read-out device reads out from the score data table 5 the following note data indicated by a value nc of the note counter 3:

nno: Note No.

leng: Length

gain: Gain

sno: Waveform data No.

eno: Envelope No.

ST3.1.2) The note data table is read out.

The note data table read-out device 6 reads out from

the note data table 7 the following parameters of a note with the note No. nno concerned:

Namly, note pitch information Tp and Tw.

ST3.1.3) A waveform data table is selected.

The waveform data selector 10 selects a waveform data table used by the note generator 11 according to the waveform data No. sno.

ST3.1.4) The envelope parameter table is read out.

The envelope parameter selector 13 selects and outputs from the envelope parameter table 12 the following parameters with the envelope No. eno concerned:

Namely, parameters AR, D1R, D2R, RR, D1L and L1M for use in determining an envelope waveform.

ST3.1.5) The kind of note is checked.

The value of the note No. nno is checked to check the kind of note to be newly played so that a corresponding operation command CMD will be set for the tone generator (SG) 11 and the envelope generator (EG) 14.

The presence of a generated tone (i.e., $1 \leq nno \leq 37$) causes a branch to 3.1.5.1.

The presence of a rest (i.e., nno == 38) causes a branch to 3.1.5.2.

The presence of a full stop (i.e., nno == 39) causes a branch to 3.1.5.3.

The operation command includes the following:

"ON" indicative of generation of a now note signal,

"OFF" indicative of termination of the generation of

the current note signal, and

"HOLD" indicative of continuation of the generation of the current note signal.

[0020]

ST3.1.5.1) In the presence of a generated tone (i.e., $1 \le nno \le 37$), the following are carried out:

Set the "ON" command for the SG 11 and EG 14, and branch to ST4.

ST3.1.5.2) In the presence of a rest (nno == 38), the following are carried out:

Set the "OFF" command for the SG11 and EG14, and branch to ST4.

ST3.1.5.3) In the presence of a full stop (nno == 39), the following are carried out:

Set the "OFF" command for the SG11 and EG14.

Set ending flag = 1.

Branch to ST4.

[0021]

ST3.2) When mc == leng * Temp - 1, the following are carried out:

Generate the final sample value of the note currently under performance.

Set the "HOLD" command for the SG 11 and EG 14.

Set measure counter mc = 0.

Set note counter nc = nc + 1.

Branch to ST4.

ST3.3) When 0 < mc < leng * Temp - 1, the following are

carried out:

Continue the generation of the note currently under performance.

Set the "HOLD" command for the SG11 and EG14.

Set measure counter mc = mc + 1.

Branch to ST4.

[0022]

ST4) The tone generator (SG) 11 generates tone data and outputs x.

Based on the operation command CMD, parameters Tp and Tw, and the waveform data table, the tone signal x as a repetitive or frequency waveform is generated at every sampling timing.

ST5) The envelope generator (EG) 14 generates an envelope waveform and outputs e.

Based on the operation command CMD, and envelope parameters AR, D1R, D2R, RR, D1L and L1M, the envelope waveform e is outputted at every sampling timing.

ST6) The output value y is calculated as follows:

y = x * e * gain

ST7) Waiting until the next sampling timing.

A sequence of processing from ST2 to ST7 is performed for every sampling period.

ST8) The generation of the melody is completed.

[0023]

Fig. 10 shows the operating state of the algorism described in Fig. 9. The melody generating controller 1 of

Fig. 1 outputs either the "ON" or "OFF" command as the operation command CMD at the first sampling timing of each note. At any other sampling timing, it outputs the "HOLD" command. The tone generator (SG) 11 of Fig. 1 outputs a repetitive or frequency signal corresponding to the interval of the note as the tone signal x. The pitch period of the repetitive signal is determined by the parameters Tp and Tw. The envelope generator (EG) 14 generates the envelope waveform e. Upon receipt of the "ON" command, the envelope generator 14 starts outputting the determined waveform. The shape of the envelope waveform is determined by the parameters AR, D1R, D2R, RR, D1L, and L1M. The multipliers 15 and 16 of Fig. 1 multiply x, e, and gain, and output the melody signal y.

[0024]

Thus, according to the 1st above-mentioned example, the score data table 5, the note data table 7, the waveform tables 8 and 9, and the envelope parameter table 12 can be memorized to ROM, and when the melody generating controller 1 controls these tables, the melody can be generated by DSP. Moreover, there is less ROM capacity used by these data tables than ROM capacity used in the conventional apparatus. Furthermore, there are few amounts of operations of processing of DSP shown in figure 9 than the conventional voice encoding and decoding method. In addition, there are the following characteristic features.

(1) By rewriting the score data table 5, change of a melody

can be performed simply.

(2) Change of timbres can be simply performed by rewriting the waveform envelope parameter tables 8, 9, and 12.

[0025]

Although the waveform data tables 8, 9 in the above-mentioned first example store plural waveforms respectively, they may store a waveform at the minimum. In this case, the waveform data selector 10 does not need providing. Further, although the envelope parameter table 12 stores plural sets of envelope parameters, it may store a set of envelope parameters at the minimum. In this case, the envelope parameter selector 13 does not need providing. Furthermore, a scale can be generated even in the case of e = 1.0 without using the envelope generator (EG) 14. In addition, if the parameters Tp and Tw are memorized to the score data table 5, the note data table read-out device 6 and the note data table 7 will not need providing.

[0026]

(Example 2)

The second example of the present invention will now be described with reference to the accompanying drawings. The general configuration of the melody generating apparatus in this example is the same as that in the first example shown in Fig. 1. A different point is that this example has another configuration of the tone generating section constituted of the waveform data tables 8, 9, the waveform selector 10, and the tone generator 11. First, the principle

of operation of the melody generating apparatus in this example will be described.

[0027]

Fig. 11 shows how a repetitive or frequency waveform is generated. In Fig. 11, h(t) represents an impulse response waveform, sp(t) represents a train of impulses whose pitch period is Tp. The bottommost repetitive waveform x(t) is determined from the following convolution equation:

$$x(t) = h(t) * sp(t),$$

where * represents the convolution operation. That is, signals shifted in time by Tp are added to obtain the repetitive waveform x(t) having the pitch period Tp. In Fig. 11, s1(t) to s6 (t) are the signals having the waveform h(t) and shifted in time by Tp, which are added to obtain the repetitive or frequency signal x(t).

[0028]

Fig. 12 shows the state of generation of a waveform having a pitch period Tp'. Here, an impulse response waveform h'(t) is determined by scaling the impulse response waveform h(t) with Tp/Tp' in the time-axis direction:

$$h'(t) = h\,(t \bullet Tp/Tp')$$

[0029]

Fig. 13 shows the state of Fig. 11 on the frequency axis. Spectra H(ω), SP(ω), and X(ω) are determined by converting time-domain signals h(t), sp(t), and x(t) to those in a frequency region. Here, X(ω) is a spectrum having a high-frequency component that is double the fundamental

frequency ωp.

[0030]

Fig. 14 shows the state of Fig. 12 on the frequency axis. $x(\omega)$ in Fig. 13 and $x'(\omega)$ in Fig. 14 have the same harmonic structure but their fundamental frequencies are different. Thus, repetitive signals identical in harmonic structure to but different in pitch period from each other can be obtained by scaling the impulse response waveform h(t) in the above-mentioned manner. This makes it possible to generate signals similar in timbre to but different in pitch period from each other, and hence to generate scales with similar timbres.

[0031]

Fig. 15 shows the configuration of the tone generating section in this example. In Fig. 15, 17 is a tone controller, 18 is a CH1 waveform data read-out device, 19 is a CH2 waveform data read-out device, 20 is an adder that adds outputs of the two waveform data read-out devices to output x, and 21 is the same waveform data table as that in Fig. 7 for storing the waveform data to be read out by the two waveform data read-out devices 18, 19.

[0032]

The tone controller 17 controls two operation CHs based on the operation command CMD and the pitch period Tp. It has the following control variables:

posp: Variable for control of the pitch period. 64 is added to the variable for every sampling period. If posp ≥

TP, posp = posp - Tp will be set, looking for a CH in a "stopped state" to turn the CH into an "operating state."

SG. stt: Variable indicative of the "stopped state" or the "operating state" when the sound source is in operation.

[0033]

The two waveform read-out devices 18, 19 read the waveform data table 21 from top to bottom in order at intervals Tw. CHn (n = 1, 2) has the following control variables:

CHn. pos: Variable indicative of the read-out position of the waveform data table 21. The waveform data table 21 is read in order while scaling it with a scaling factor of Tw. After completion of the reading of the waveform data table 21, the operating state CHn. stt is changed to the "stopped state."

SG. stt: Variable indicative of the state of operation of each CH, that is, either the "operating state" or the "stopped state."

[0034]

Fig. 16 shows the state in which the two waveform data read-out devices 18, 19 read the waveform data table 21. They perform the reading of the waveform data table 21 while scaling it with the parameter Tw, respectively. Here, two CHs are used and respective outputs become x1 and x2.

[0035]

The operation of the tone generation section will be described below with reference to Figs. 17 to 20. Processing

shown in Fig. 17 is for [SGEN] processing in ST4 of Fig. 9.

[0036]

ST11) The sound source CHs are controlled.

Processing [SGCHCNT] in Fig. 17 is shown in Fig.18; it corresponds to the operation of the tone controller 17 in Fig. 15.

ST11.1) The operation command CMD is checked.

Operation command CMD == "OFF" causes a branch to ST11.1.1.

Operation command CMD == "ON" causes a branch to ST11.1.2.

Operation command CMD == "HOLD" causes a branch to ST11.1.3.

ST11.1.1) When the operation command == "OFF," the following are carried out:

Set the state of operation of the CH1 as CH1. stt = "stopped state."

Set the state of operation of the CH2 as CH2. stt = "stopped state."

Set the state of the sound source as SG. stt = "stopped state."

Branch to ST12.

ST11.1.2) When the operation command CMD == "ON," the following are carried out:

Initialize the CH1 and CH2.

Set the state of operation of the CH1 as CH1. stt = "stopped state."

Set the state of operation of the CH2 as CH2. stt = "stopped state."

Set the state of the sound source as SG. stt = "operating state."

Branch to ST12.

ST11.1.3) When the operation Command CMD== "HOLD," the following are carried out:

Check the state of the sound source SG. stt.

Branch to ST11.1.3.1 when the sound source SG. stt. = "operating state," or

Branch to ST2 when the sound source SG. stt. = "stopped state."

ST11.1.2) When the operation command is in the "operating state," the following are carried out:

Set posp = posp + 64, and

if posp ≥ Tp, carry out the following:

Set posp = posp - Tp.

Look for a CH in the stopped state to assign a number n to the CH. Then, set an initial value for CHn. pos to turn the CH into the "operating state."

Set CHn. pos = (int) (posp * Tp/64).

Turn the CHn. stt into the "operating state."

[0037]

ST12) The sound source CH1 is operated.

Processing [SGH10OR] in Fig. 17 is shown in Fig. 19; it corresponds to the operation of the CH1 waveform data read-out device 18 in Fig. 15.

ST12.1) The state of operation of the CH 1, CH1. stt is checked as follows:

CH1. stt = "operating state" causes a branch to ST12.1.1.

CH1. stt = "stopped state" causes a branch to ST12.1.2.

ST12.1.1) When CH1.stt = "operating state," the value of CH1. pos is checked as follows:

CH1. pos ≥ TL causes a branch to ST12.1.1.1; or

CH1. pos < TL causes a branch to ST12.1.1.2.

ST12.1.1.1) When CH1. pos ≥ TL, the reading of the data table 21 is completed, and the following are carried out:

Set x1 = 0.

Set the state of operation of the CH 1 as CH1. stt = "stopped state."

Branch to ST13.

ST12.1.1.2) When CH1. pos < TL, the following are carried out:

Read data at the position of CH1. pos in the waveform data table to output the result to x1.

Set CH1. pos = CH1. pos + Tw.

Branch to ST13.

ST12.1.2) When CH1.stt == "stopped state," the following are carried out:

Set x1 = 0.

Branch to ST13.

[0038]

ST13) The CH2 is operated.

Processing [SGH2OPR] in ST13 of Fig. 17 is shown in Fig. 20; it corresponds to the operation of the CH2 waveform data read-out device 19 in Fig. 15. The CH2 waveform data read-out device 19 operates in the same manner as the CH1 waveform data read-out device 18 to output the result to x2.

[0039]

ST14) The outputs x1 and x2 of the CH1 and CH2 are added at the adder 20 as follows:

x = X1 + X2

[0040]

Fig. 21 shows how the two waveform data read-out devices 18, 19 of Fig 15 generate the repetitive or frequency waveform x(t) having the period Tp.

[0041]

Fig. 22 shows an operation example of the tone generating section when Tp = 100 and Tw = 10. Here, h[i] indicates a waveform data table. In the CH1 waveform data read-out device 18, the initial value of the read-out position of the waveform data table h[i] is 0, and the read-out interval is Tw = 10. In other words, data at positions of i= 0, 10, 20,… are outputted as x1 sequentially at every sampling timing. In the CH2 waveform data read-out device 19, the following value is the initial value of the read-out position of the data table:

(int)((128 - Tp)/64 * Tw) = 4

Therefore, the CH2 sequentially outputs data as x2 at positions of i = 4, 14, 24, … in the waveform data table h[i].

[0042]

The execution of the above-mentioned processing enables the setting of the pitch period Tp of output signals from the tone generating section with an accuracy of 1/64 of the sampling period. If the sampling frequency fs = 8kHz, the pitch period Tp will need to be set with such a degree of accuracy so as to generate the pitch of an accurate scale. The value Tp for each note is calculated as follows:

$$Tp = fs \times 64/f,$$

where fs is the sampling frequency [Hz], and
f is the pitch frequency of the note [Hz].

For example, for A4 note, if f = 440 [Hz] and fs = 8000 [Hz], then Tp = 1164. Further, the value Tw in the note data table of Fig. 6 is calculated from the following equation:

$$Tw = f \times 16384/fs,$$

where 16384 is an arbitrary constant. It is assumed here that the size of the waveform data table is 16384 and the constant is so determined that the read-out of the waveform data table will be completed within the pitch period. Although the size of the waveform data table was fixed to 16384, the size of an actual waveform data table can be set to 256 using interpolation of data by primary approximation.

[0043]

As described above and according to the second example, the contents of the waveform data table 21 are arbitrary and a repetitive or frequency signal having the pitch period Tp given as a parameter can be generated regardless of the

contents. Therefore, repetitive signals of various harmonic structures can be generated depending on the contents of the waveform data table 21, which makes it possible to produce a variety of timbres. Further, the amounts of operations on the DSP shown in Figs. 17 to 20 may be reduced.

[0044]

(Example 3)

The third example of the present invention will now be described with reference to the accompanying drawings. This example shows another operation example of the envelope generator (EG) 14 shown in Fg. 1.

[0045]

Fig. 23 shows an algorism for the operation of the envelope generator (EG) in this example. Processing of Fig. 23 is for processing [EGEN] in ST5 of Fig. 5. The envelope generator (EG) starts generating an envelope waveform in response to the "ON" command. The shape of the waveform is determined by the parameters AR, D1R, D2R, D1L, RR, and LIM.

[0046]

Fig. 24(a) shows an envelope waveform e caused by the "ON" command. Fig. 24(b) shows an envelope waveform e caused by receiving the "OFF" command halfway through the generation of the envelope waveform.

[0047]

Fig. 25 shows a state transition diagram for generation of the envelope waveform. In each state of Fig. 25, the envelope generator (EG) operates as follows:

1) "AR State"

The leading edge of the envelope waveform e is generated. The rise speed is determined by AR. If $e \geq 1.0$, a transition to a "D1R state" will occur as $e = 1.0$. Fig. 26 shows the operation algorism. In the "AR state," e changes at every sampling timing as follows:

$$e = e + (2.0 - e) * AR$$

2) "D1R state"

The first leading edge of the envelope waveform e is generated. The rise speed is determined by D1R. If $e \leq D1L$, a transition to a "D2R state" will occur as $e = D1L$. Further, if e becomes as $e \leq LIM$ on its way, a transition to the "OFF state" will occur as $e = 0$. Fig 27 shows the operation algorism. In the "D1R state," e changes at every sampling timing as follows:

$$e = e * D1R$$

3) "D2R state"

The second leading edge of the envelope waveform e is generated. The rise speed is determined by D2R. If $e \leq LIM$, a transition to the "OFF state" will occur as $e = 0$. Fig. 28 shows the operation algorism. In the "D2R state," e changes at every sampling timing as follows:

$$e = e * D2R$$

4) "RR state"

When the "OFF" command is received in any one of the "AR state," the "D1R state," and the "D2R state," a transition to an "RR state" will occur. This state generates

the falling edge of the envelope waveform e and the fall speed is determined by RR. If e ≤ L1M, a transition to the "OFF state" will occur as e = 0. Fig. 29 shows the operation algorism. In the "RR state," e changes at every sampling timing as follows:

$$e = e * RR$$

5) "OFF state"

e = 0 is set.

[0048]

In the envelope parameter table of Fig. 8, a parameter having envelope No. 0 causes a continuous sound, and a parameter having envelope No. 1 or 2 causes an attenuated sound.

[0049]

As described above and according to the third example, an excellent envelope generator capable of generating an envelope waveform according to a simple algorism using the parameters AR, D1R, D2R, RR, D1L, and LIM can be realized.

[0050]

[Effects of the Invention]

It is apparent from the first example that the present invention has the following effects:

1) Since a melody is stored in the form of a score, required ROM capacity can be reduced.

2) Since the conventional coding/decoding method is not used, the amounts of required operations on the DSP can be reduced.

3) The score data table can be rewritten to easily change or alter the melody.

4) The waveform data table and the envelope parameter table can also be rewritten to easily change timbres.

[0051]

It is also apparent from the second example that the present invention has the following effects:

1) Tones of many timbres can be produced according to the contents of the waveform data table.

2) The pitch period can be set finer than the sampling period.

3) It can be realized by DSP using a simple algorism.

[0052]

It is further apparent from the third example that the present invention has the following effects:

1) An envelope waveform can be generated by DSP using a simple algorism.

[Brief Description of the Drawings]

[Fig. 1] It is a block diagram of a melody generating apparatus by DSP according to the first example of the present invention.

[Fig. 2] It is a list showing a score data table in the first example of the present invention.

[Fig. 3] It is a list showing a concrete example of the score data table in the first example of the invention.

[Fig. 4] It is a diagram showing a musical score upon which the concrete example of the score data table in the

first example of the present invention is based.

[Fig. 5] It is a list of a note data table in the first example of the present invention.

[Fig. 6] It is a list showing a concrete example of a note data table in the first example of the present invention.

[Fig. 7] It is a waveform chart showing an example of a waveform data table in the first example of the present invention.

[Fig. 8] It is a list showing an example of an envelope parameter table in the first example of the present invention.

[Fig. 9] It is a flowchart showing an algorithm in the first example of the present invention.

[Fig. 10] It is a waveform chart showing the operation in the first example of the present invention.

[Fig. 11] It is a timing chart for explaining the principle of operation in a time domain of a tone generator in the second example of the present invention.

[Fig. 12] It is a timing chart for explaining the principle of operation in a time domain of the tone generator in the second example of the present invention.

[Fig. 13] It is a timing and spectral chart for explaining the principle of operation in a frequency region of the tone generator in the second example of the present invention.

[Fig. 14] It is a timing and spectral chart for explaining the principle of operation in a frequency region of the tone generator in the second example of the present

invention.

[Fig. 15] It is a block diagram of tone generating means in the second example of the present invention.

[Fig. 16] It is a timing chart for reading out a waveform data table in the second example of the present invention.

[Fig. 17] It is a flowchart showing an algorithm for the tone generating means in the second example of the present invention.

[Fig. 18] It is a flowchart showing the algorithm for the tone generating means in the second example of the present invention.

[Fig. 19] It is a flowchart showing the algorithm for the tone generating means in the second example of the present invention.

[Fig. 20] It is a flowchart showing the algorithm for the tone generating means in the second example of the present invention.

[Fig. 21] It is a timing chart for explaining the operation of the tone generating means in the second example of the present invention.

[Fig. 22] It is a timing chart for explaining the operation of the tone generating means in the second example of the present invention.

[Fig. 23] It is a flowchart showing an algorithm for envelope generating means in the third example of the present invention.

[Fig. 24] It is a waveform chart in the third example of the present invention.

[Fig. 25] It is a state transition diagram of the envelope generating means in the third example of the present invention.

[Fig. 26] It is a flowchart showing an algorithm for the envelope generating means in the third example of the present invention.

[Fig. 27] It is a flowchart showing the algorithm for the envelope generating means in the third example of the present invention.

[Fig. 28] It is a flowchart showing the algorithm for the envelope generating means in the third example of the invention.

[Fig. 29] It is a flowchart showing the algorithm for the envelope generating means in the third example of the invention.

[Fig. 30] It is a block diagram showing the general configuration of a conventional melody generating apparatus.

[Fig. 31] It is a block diagram of the melody generating apparatus by conventional DSP.

[Brief description marks]

1: melody generating controller

2: measure counter

3: note counter

4: score data table read-out device

5: score data table

6: note data table read-out device

7: note data table

8: waveform data table

9: waveform data table

10: waveform data selector

11: tone generator (SG)

12: envelope parameter table

13: envelope parameter selector

14: envelope generator (EG)

15: multiplier

16: multiplier

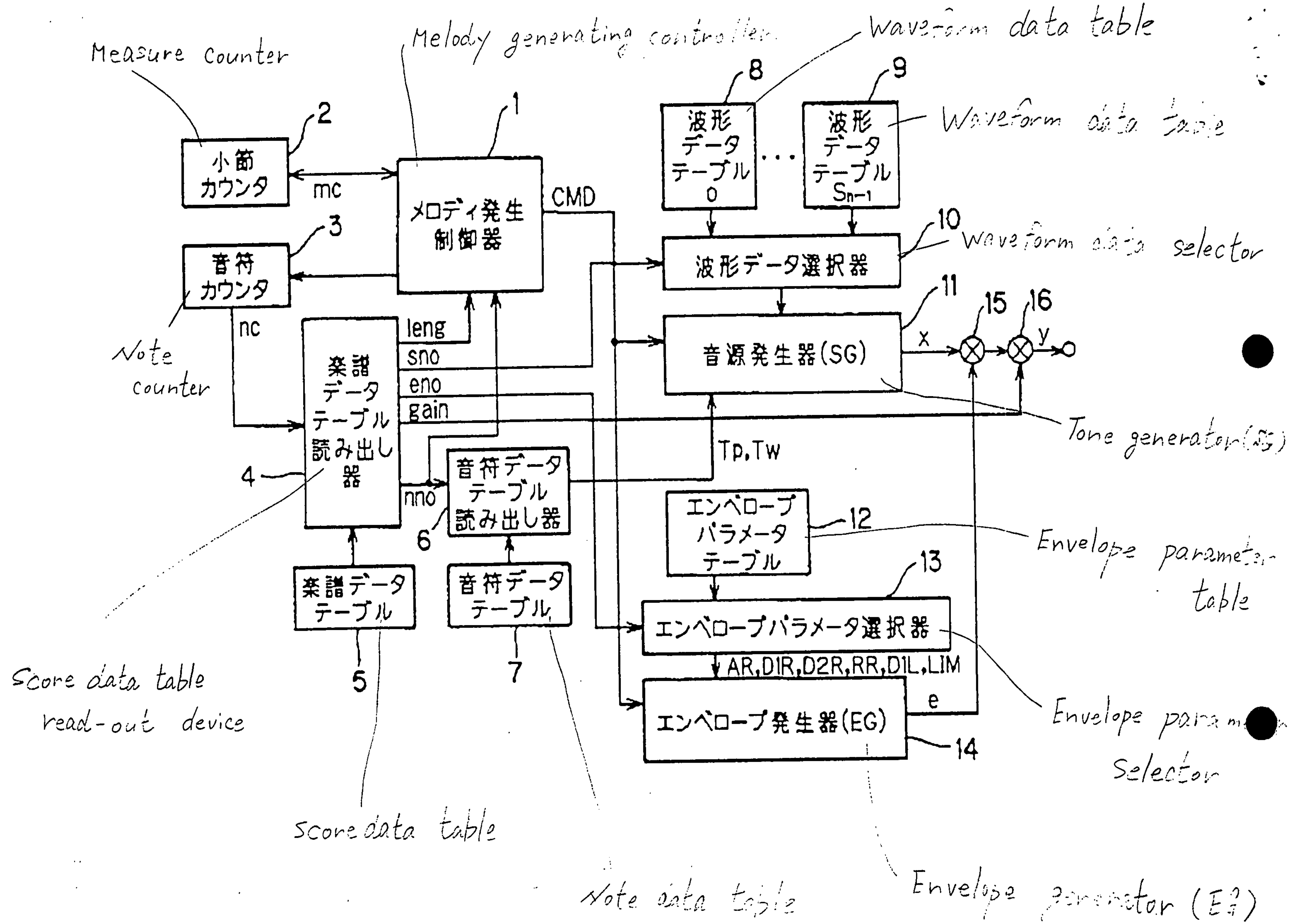
17: tone controller

18: CH1 waveform data read-out device

19: CH2 waveform data read-out device

20: adder

21: waveform data table


Measure counter
Melody generating controller
Waveform data table
小節
カウンタ
2
mc
メロディ発生
制御器
1
CMD
音符
カウンタ
3
nc
Note counter
楽譜
データ
テーブル
読み出し
器
4
leng
sno
eno
gain
nno
音符データ
テーブル
読み出し器
6
楽譜データ
テーブル
5
音符データ
テーブル
7
Score data table read-out device
Score data table
Note data table
波形
データ
テーブル
0
8
波形
データ
テーブル
S_{n-1}
9
Waveform data table
波形データ選択器
10
Waveform data selector
音源発生器(SG)
11
x
15
16
y
Tone generator (SG)
Tp,Tw
エンベロープ
パラメータ
テーブル
12
Envelope parameter table
エンベロープパラメータ選択器
13
AR,D1R,D2R,RR,D1L,LIM
エンベロープ発生器(EG)
14
e
Envelope parameter selector
Envelope generator (EG)

【図2】

Serial number / Note name / Note No. / Length / Gain / Waveform data No. / Envelope No.

通し番号	音符名	音符No. nno	長さ leng	ゲイン gain	波形データNo. sno	エンベロープNo. eno
⋮	⋮	⋮	⋮	⋮	⋮	⋮

【図5】

Note No. / Note name.

音符No.	音符名	Tp	Tw
⋮	⋮	⋮	⋮

【図8】

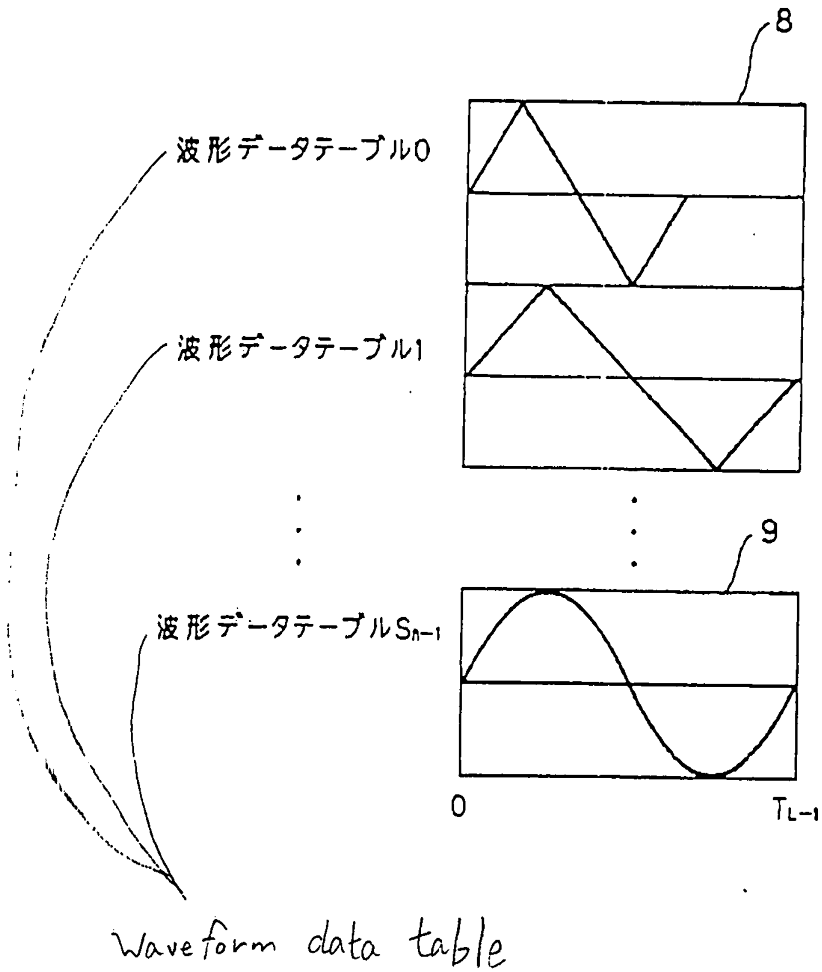
Envelope No.

エンベロープNo.	AR	D1R	D2R	RR	D1L	LIM
0	1.00	1.00	1.00	0.00	0.00	0.02
1	0.10	0.95	0.999	0.98	0.50	0.02
2	0.10	0.90	0.999	0.95	0.70	0.02

【図１７】

[SGEN]
音源データを発生する — Generate tone data
ST11 音源CHの制御を行う[SGCHCNT] Control sound source CH
ST12 音源CH1の制御を行う[SGCH1OPR] Control sound source
ST13 音源CH2の制御を行う[SGCH2OPR] Control sound source CH2
ST14 $x = x1 + x2$

【図7】



【図29】

[EGRROPR]

RR状態の動作を行う Operate in PR state

e = e * RR

eの値がLIM以下の時"OFF状態"に遷移する

(e ≦ LIM)

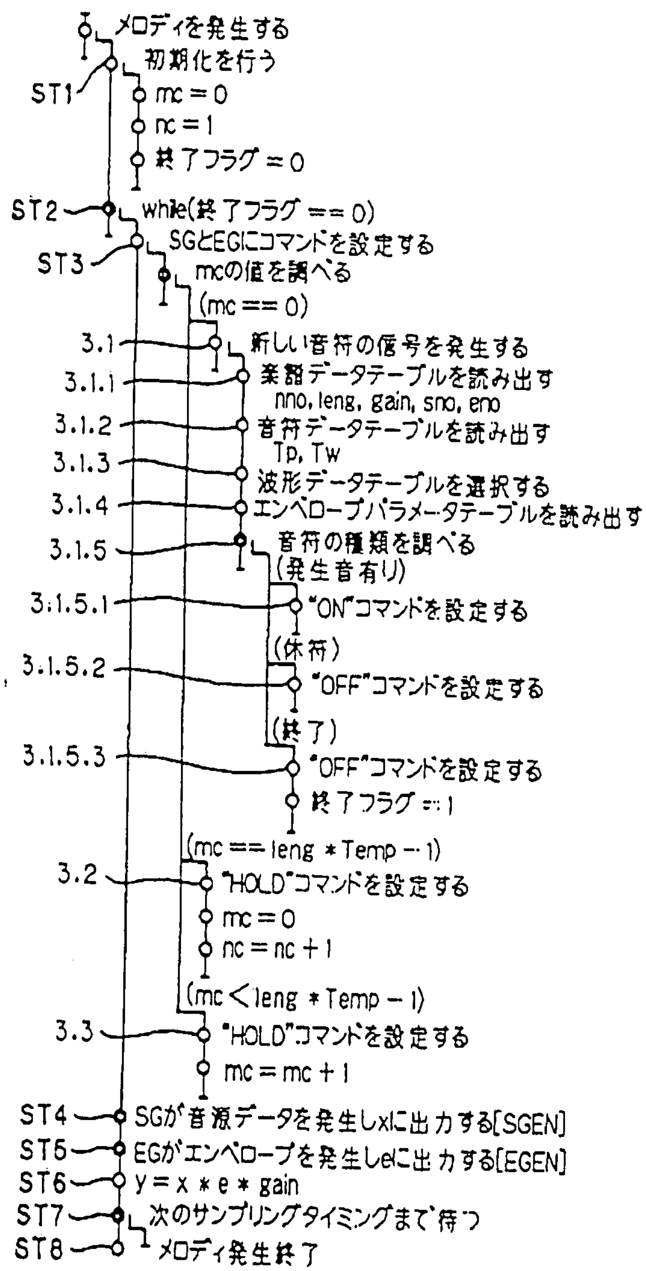
e = 0

EG.stt = "OFF状態" EG.stt = "OFF state"

(e > LIM)

Transition to "OFF state" occur if value e is LIM or less

【図9】



Generate melody

ST1 Initialization

Ending flag = 0

ST2 (Ending flag == 0)

ST3 Set command for SG and EG

Check value mc

3.1 Generate new note signal

3.1.1 Read out score data table

3.1.2 Read out note data table

3.1.3 Select wave data table

3.1.4 Read out envelope parameter table

3.1.5 Check kind of note

(Presence of generated sound)

3.1.5.1 Set "ON" command

(Rest)

3.1.5.2 Set "OFF" command

(Full stop)

3.1.5.3 Set "OFF" command

Ending flag = 1

3.2 Set "HOLD" command

3.3 Set "HOLD" command

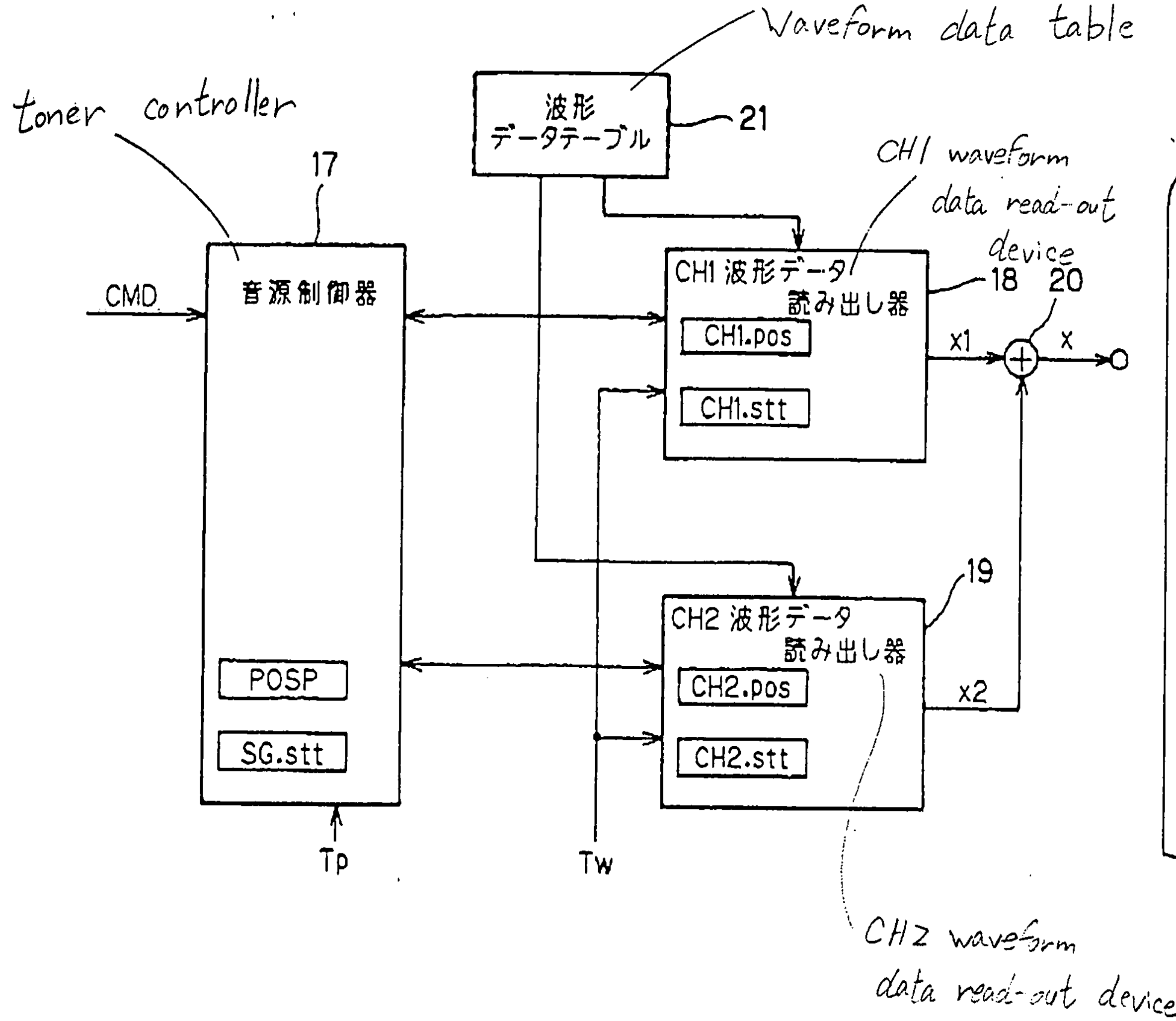
ST4 SG generates and outputs tone data to x

ST5 EG generates and outputs envelope to e

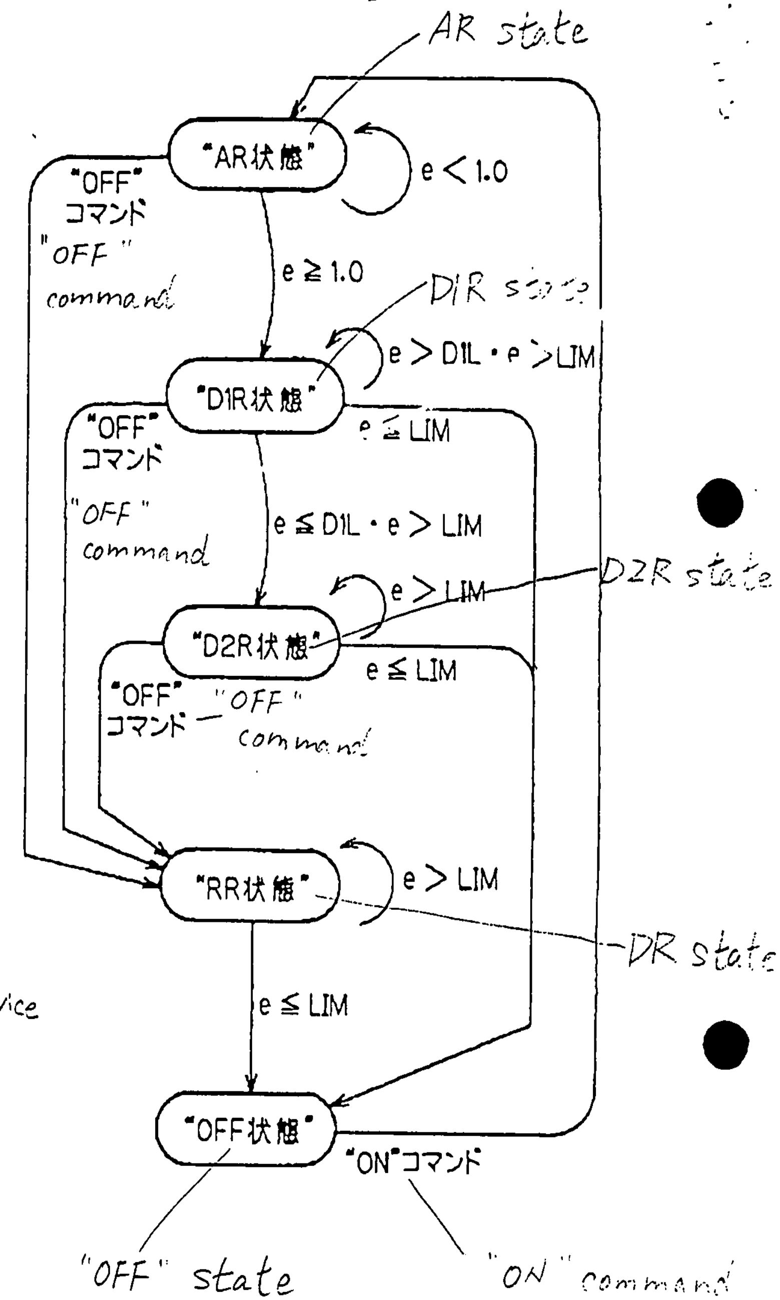
ST7 Wait until next sampling timing

ST8 End of melody generation

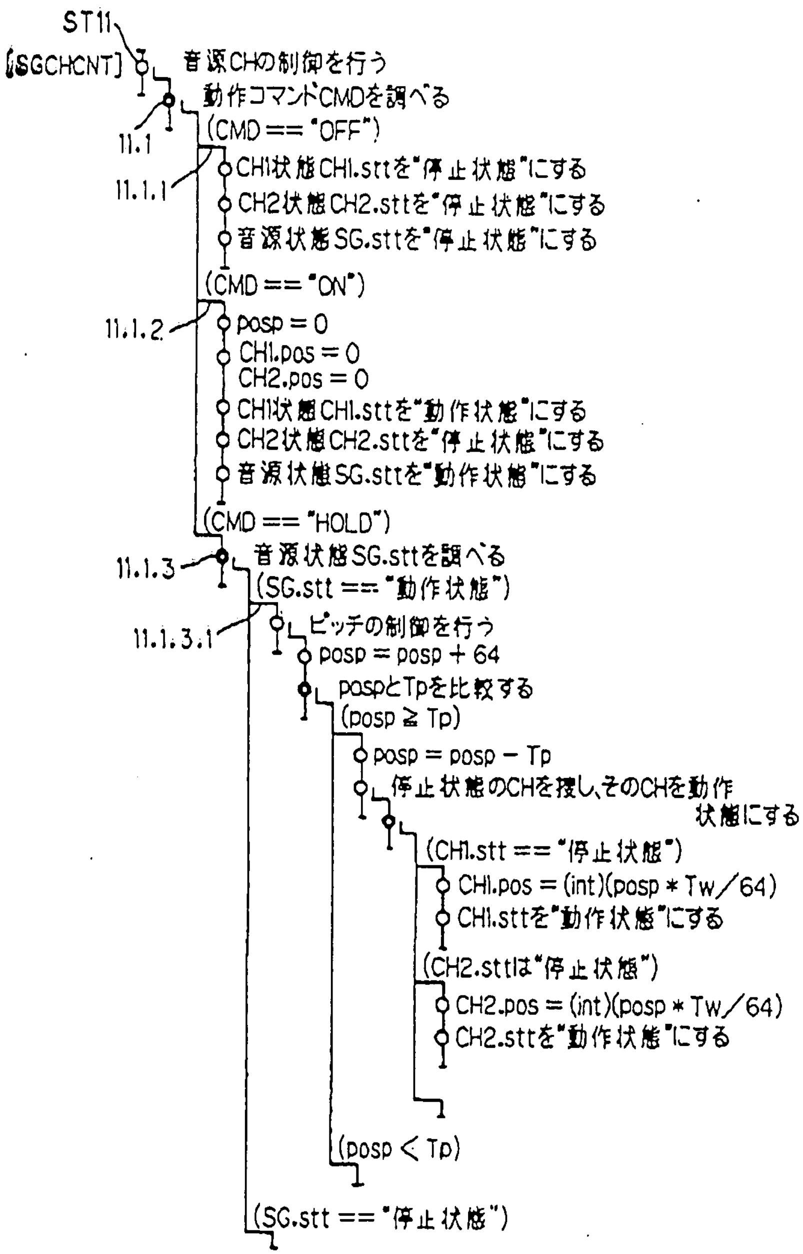
【図１５】


Waveform data table
波形
データテーブル
21
toner controller
17
CMD
音源制御器
POSP
SG.stt
Tp
CH1 waveform
data read-out
device
CH1 波形データ
読み出し器
CH1.pos
CH1.stt
18
20
x1
x
CH2 波形データ
読み出し器
CH2.pos
CH2.stt
19
x2
Tw
CH2 waveform
data read-out device

【図２５】


AR state
"AR状態"
e < 1.0
"OFF"
コマンド
"OFF"
command
e ≧ 1.0
D1R state
"D1R状態"
e > D1L・e > LIM
e ≦ LIM
"OFF"
コマンド
"OFF"
command
e ≦ D1L・e > LIM
"D2R状態"
e > LIM
D2R state
e ≦ LIM
"OFF"
コマンド
"OFF"
command
"RR状態"
e > LIM
DR state
e ≦ LIM
"OFF状態"
"ON"コマンド
"OFF" state
"ON" command

【図１８】



ST11 Control sound source CH

11.1 Check operation command CMD

11.1.1 Turn CH1 state CH1. stt into "stopped state"

Turn CH2 state CH2. stt into "stopped state"

Sound source state SG. stt into "stopped state"

11.1.2 Turn CH1 state CH1. stt into "operating state"

Turn CH2 state CH2. stt into "stopped state"

Sound source state SG. stt into "operating state"

11.1.3 Check sound source state SG. stt

(SG. stt == "operating state")

11.1.3.1 Control pitch

Compare pose and Tp

Look for CH in stopped state to turn CH into operating state

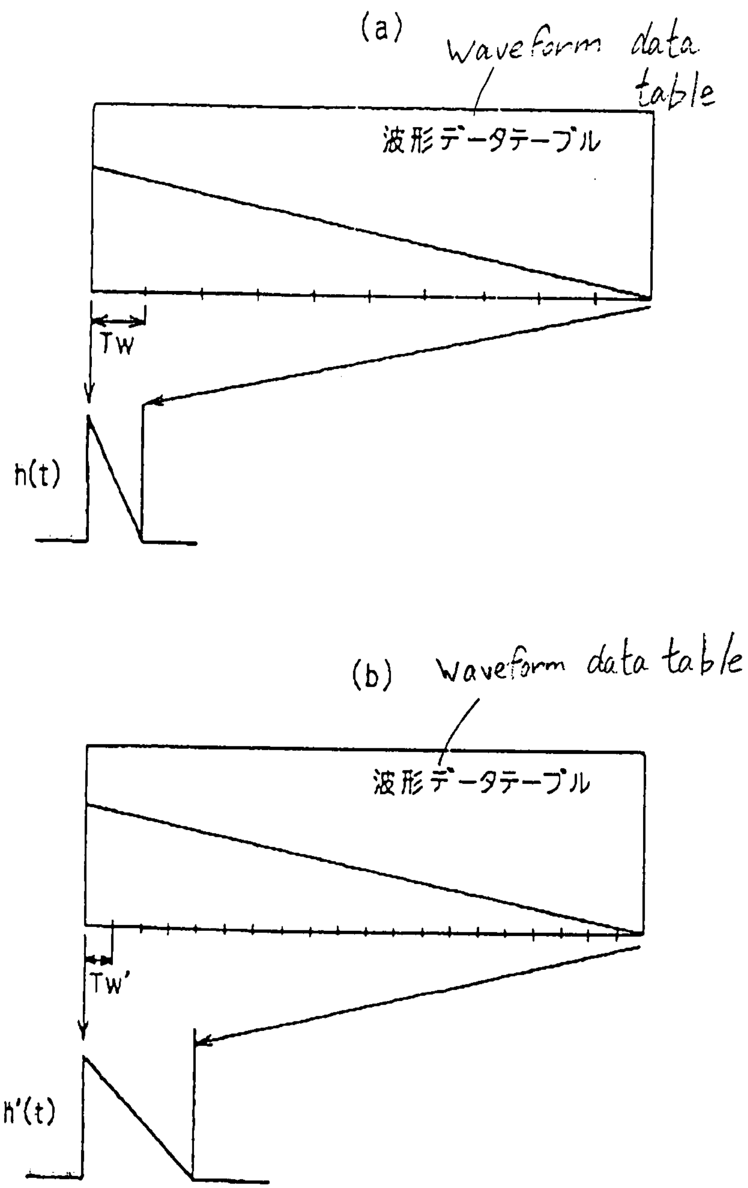
Turn CH1. stt into "operating state"

(CH2. stt is in "stopped state")

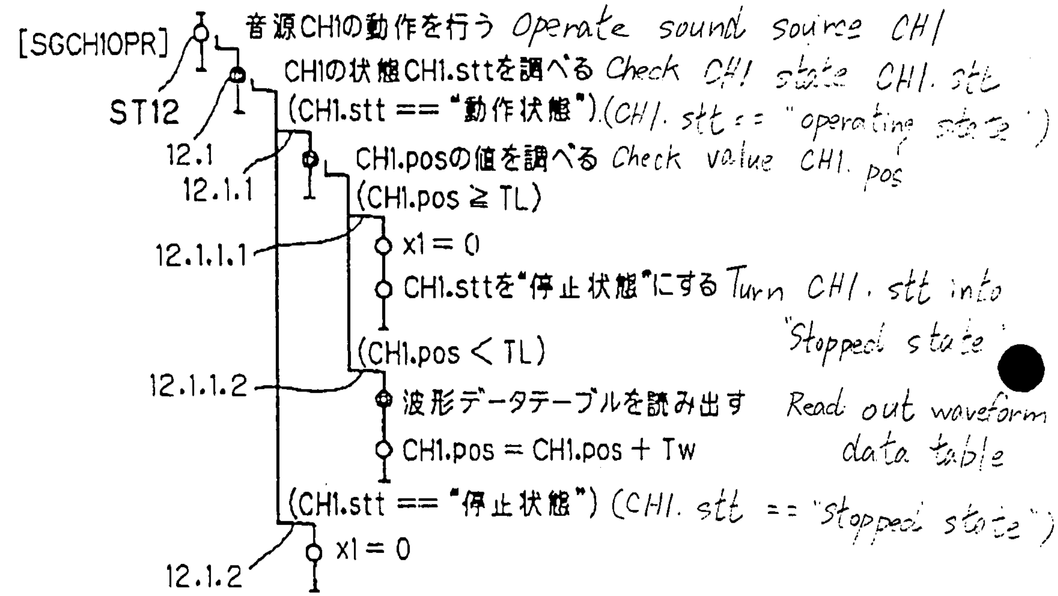
Turn CH2. stt into "operating state"

(SG. stt == "stopped state")

【図16】


(a)
Waveform data table
波形データテーブル
Tw
h(t)
(b)
Waveform data table
波形データテーブル
Tw'
h'(t)

【図19】


[SGCH1OPR]
ST12
音源CH1の動作を行う Operate sound source CH1
12.1
CH1の状態CH1.sttを調べる Check CH1 state CH1.stt
(CH1.stt == "動作状態") (CH1.stt == "operating state")
12.1.1
CH1.posの値を調べる Check value CH1.pos
(CH1.pos ≧ TL)
12.1.1.1
x1 = 0
CH1.sttを"停止状態"にする Turn CH1.stt into "Stopped state"
(CH1.pos < TL)
12.1.1.2
波形データテーブルを読み出す Read out waveform data table
CH1.pos = CH1.pos + Tw
(CH1.stt == "停止状態") (CH1.stt == "stopped state")
12.1.2
x1 = 0

【図２０】

[SGCH20PR] ST13

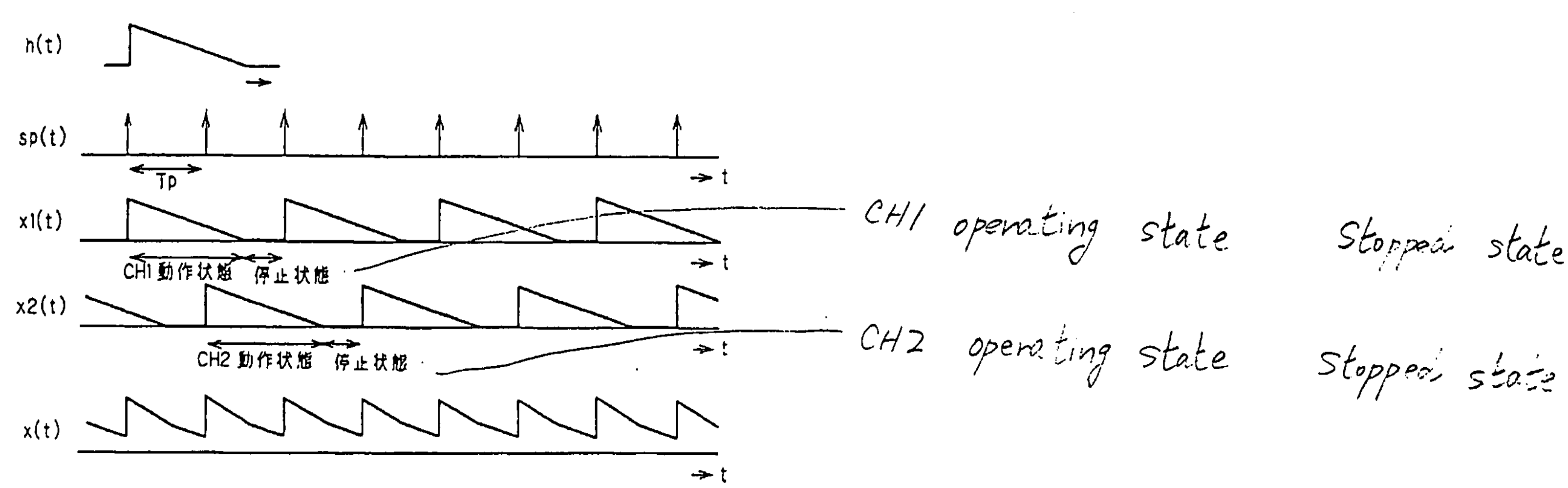
- 音源CH2の動作を行う *Operate sound source CH2*
  - CH2の状態CH2.sttを調べる *Check CH2 state CH2. stt*
    - (CH2.stt == "動作状態") *(CH2.stt == "Operating state")*
      - CH2.posの値を調べる *Check value CH2. pos*
        - (CH2.pos ≧ TL)
          - x2 = 0
          - CH2.sttを"停止状態"にする *Turn CH2. stt into "stopped state"*
        - (CH2.pos ＜ TL)
          - 波形データテーブルを読み出す *Read out waveform data table*
          - CH2.pos = CH2.pos + Tw
    - (CH2.stt == "停止状態") *(CH2.stt == "Stopped state")*
      - x2 = 0

【図２６】

[EGAROPR]

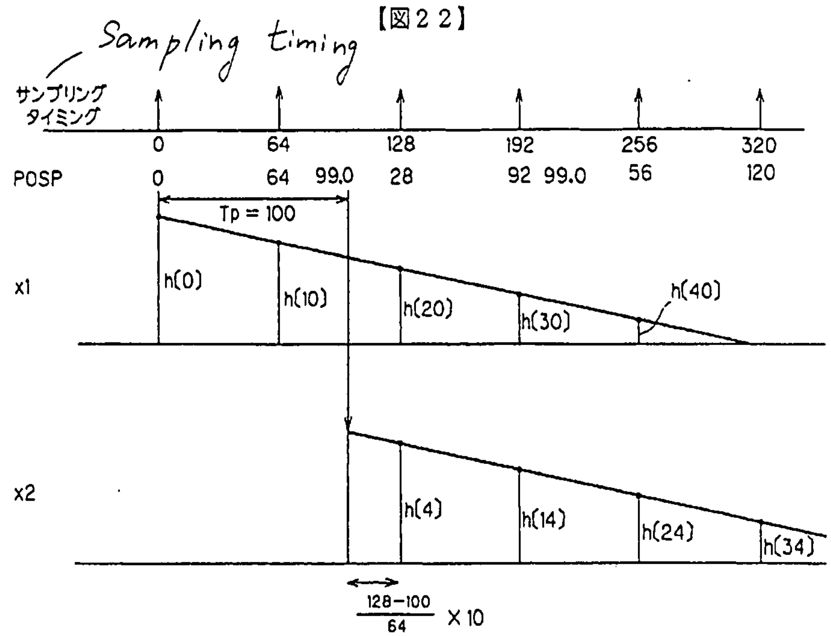
- AR状態の動作を行う *Operate in AR state*
  - e = e + (2.0 − e) * AR
  - eの値が1以上の時"D1R状態"に遷移する *Transition to "D1R state" occur if value e is 1 or more*
    - (e ≧ 1.0)
      - e = 1.0
      - EG.stt = "D1R状態" *EG. stt = "D1R state"*
    - (e ＜ 1.0)

【図21】



h(t)
sp(t)
Tp
x1(t)
CH1 動作状態
停止状態
x2(t)
CH2 動作状態
停止状態
x(t)
t
CH1 operating state
Stopped state
CH2 operating state
Stopped state

【図22】



Sampling timing
サンプリング
タイミング
0
64
128
192
256
320
POSP
0
64
99.0
28
92
99.0
56
120
Tp = 100
x1
h(0)
h(10)
h(20)
h(30)
h(40)
x2
h(4)
h(14)
h(24)
h(34)
$\frac{128-100}{64} \times 10$

【図２３】

```
[EGEN] エンベロープを発生する
  エンベロープ発生器EGの動作を動作コマンドCMDにより制御する
    動作コマンドCMDを調べる
      (CMD == "OFF")
        EGの動作状態G.sttを"RR状態"にする
      (CMD == "ON")
        EGの動作状態G.sttを"AR状態"にする
      (CMD == "HOLD")
  エンベロープを発生する
    EGの動作状態EG.sttを調べる
      (EG.stt == "OFF状態")
        e = 0
      (EG.stt == "AR状態")
        AR状態の動作を行う[EGAROPR]
      (EG.stt == "D1R状態")
        D1R状態の動作を行う[EGD1ROPR]
      (EG.stt == "D2R状態")
        D2R状態の動作を行う[EGD2ROPR]
      (EG.stt == "RR状態")
        RR状態の動作を行う[EGRROPR]
```

Generate envelope

Control operation of envelope generator EG according to operation command CMD

Check operation command CMD

Turn EG operation state G. stt into "RR state"

Turn EG operation state G. stt into "AR state"

Generate envelope

Check EG operation state EG. stt

(EG. stt == "OFF state")

(EG. stt == "AR state")

operate in AR state

(EG. stt == "D1R state")

Operate in D1R state

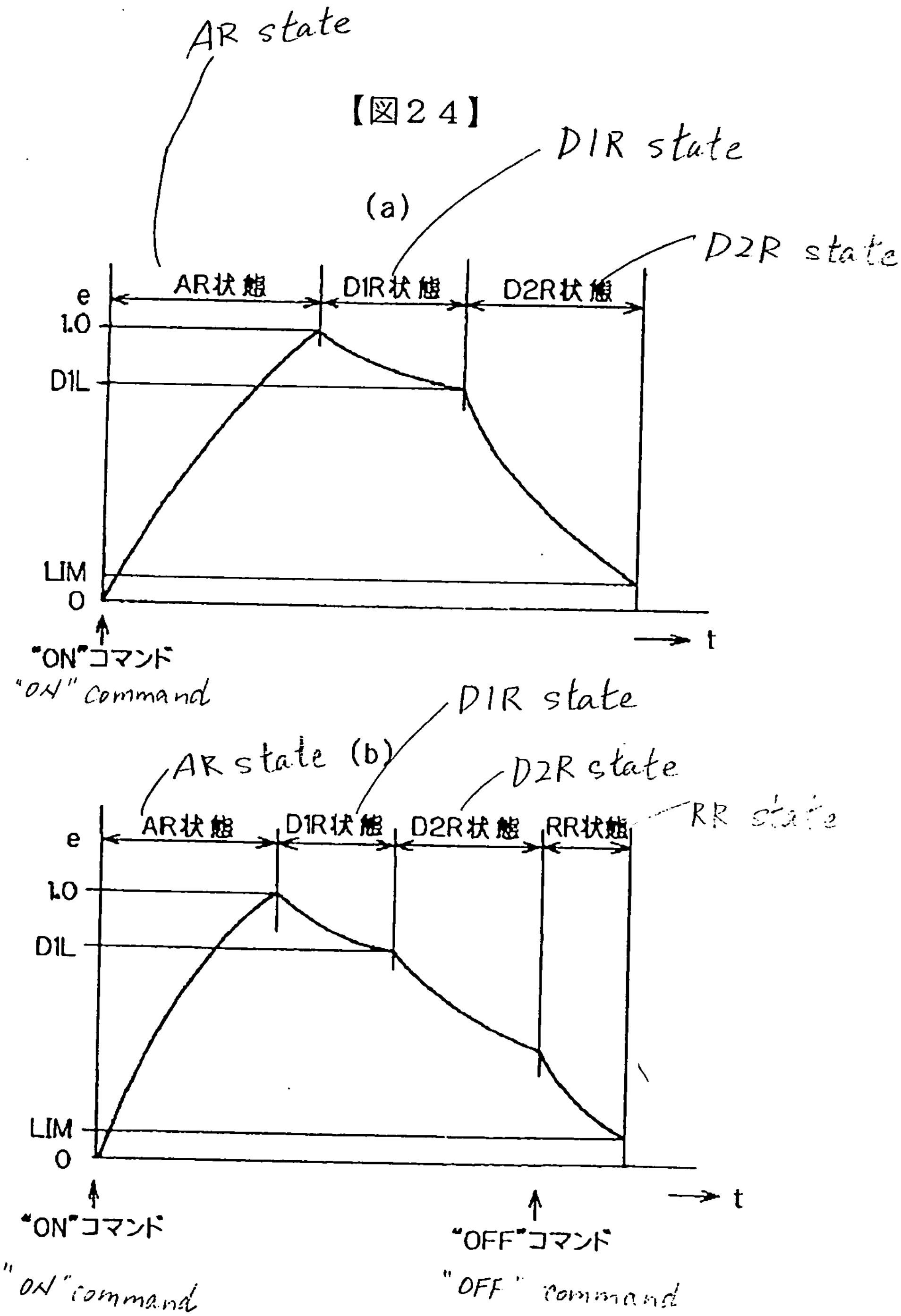
(EG. stt == "D2R state")

Operate in D2R state

(EG. stt == "RR state")

Operate in RR state

【図２４】



AR state
D1R state
D2R state
(a)
AR状態
D1R状態
D2R状態
e
1.0
D1L
LIM
0
t
"ON"コマンド
"ON" command
(b)
AR state
D1R state
D2R state
RR state
AR状態
D1R状態
D2R状態
RR状態
e
1.0
D1L
LIM
0
t
"ON"コマンド
"ON" command
"OFF"コマンド
"OFF" command

【図２７】

[EGD1ROPR] D1R状態の動作を行う *Operate in D1R state*

e = e * D1R

eの値がLIM以下の時"OFF状態"に遷移する

(e ≦ LIM)

e = 0

EG.stt = "OFF状態" *EG. stt = "OFF state"*

(e > LIM)

eの値がD1L以下の時"D2R状態"に遷移する

(e ≦ D1L)

e = D1L

EG.stt = "D2R状態" *EG stt = "D2R state"*

(e > D1L)

Transition to "OFF state" occur if value e is LIM or less

Transition to "D2R state" occur if value e is D1R or less

【図２８】

[EGD2ROPR] D2R状態の動作を行う *Operate in D2R state*

e = e * D2R

eの値がLIM以下の時"OFF状態"に遷移する *Transition to "OFF state occur if value e is LIM or less*

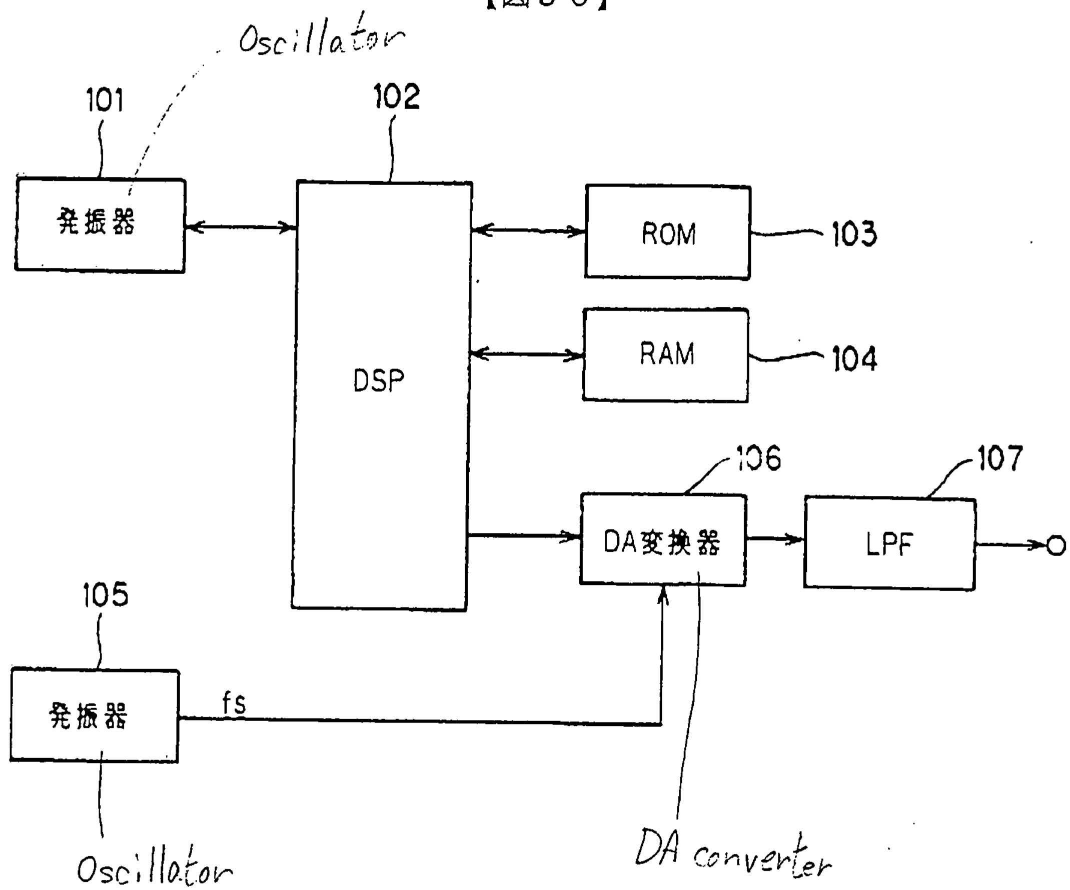
(e ≦ LIM)

e = 0

EG.stt = "OFF状態" *EG stt = "OFF state"*

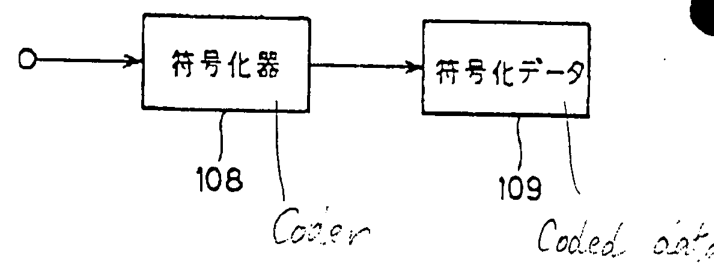
(e > LIM)

【図３０】

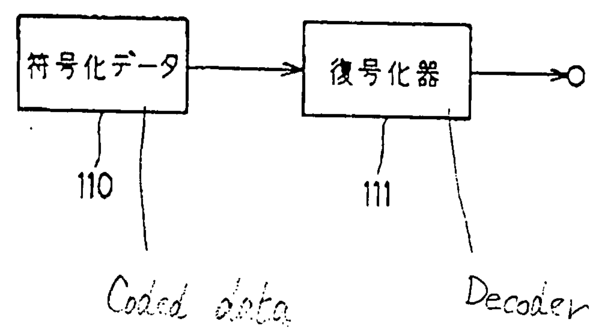

Oscillator
101
102
発振器
DSP
ROM
103
RAM
104
106
107
DA変換器
LPF
105
発振器
fs
Oscillator
DA converter

【図３１】

(a)


符号化器
符号化データ
108
109
Coder
Coded data

(b)


符号化データ
復号化器
110
111
Coded data
Decoder